NEWTON EXAMPLES

MAPLE

```
import math as m
import matplotlib.pyplot as plt

k = 0.005
g = 9.8
th = 60*(m.pi/180)
v_0 = 600
u = v_0*m.cos(th)
v = v_0*m.sin(th)
t = 0
y = (-g)*(t/k)+((k*v+g)/(k**2))*(1-m.exp((-k)*t))

while t <= 120 and y>=0:
   x = (u/k)*(1-m.exp((-k*t)))
   y = (-g)*(t/k)+((k*v+g)/(k**2))*(1-m.exp((-k)*t))
   x = round(x,2)
   y = round(y,2)
   print(x,y)
   t += 1
   plt.plot(x,y)
   plt.show()
```

Code for finding the roots of the transcendental equation:

```
> t1 := {T = (k*V+g)*(1-exp(-k*T))/(g*k)};
```

```
> k := 0.1e-6;

> V := 600*sin(3.14159265358979*(1/3));

> g := 9.8;
fsolve(t1,{T=10..120});

                    {T = 106.0398126}


> t2 := {T = (a*V+g)*(1-exp(-a*T))/(g*a)};
> a := 0.005;

> V := 600*sin(3.14159265358979*(1/3));
> g := 9.8;

> fsolve(t2, {T = 10 .. 120});

                    {T = 98.06233448}


> t3 := {T = (b*V+g)*(1-exp(-b*T))/(g*b)};
> b := 0.01;

> V := 600*sin(3.14159265358979*(1/3));
> g := 9.8;

> fsolve(t3, {T = 10 .. 120});

                    {T = 92.10191668}
```

and so on for the various values of k.

MATLAB

```
import math as m
import matplotlib.pyplot as plt

k = 0.005
g = 9.8
th = 60*(m.pi/180)
v_0 = 600
u = v_0*m.cos(th)
v = v_0*m.sin(th)
t = 0
y = (-g)*(t/k)+((k*v+g)/(k**2))*(1-m.exp((-k)*t))

while t <= 120 and y>=0:
    x = (u/k)*(1-m.exp((-k*t)))
    y = (-g)*(t/k)+((k*v+g)/(k**2))*(1-m.exp((-k)*t))
    x = round(x,2)
    y = round(y,2)
    print(x,y)
    t += 1
    plt.plot(x,y)
    plt.show()
```

Code for finding the roots of the transcendental equation:

```
> t1 := {T = (k*V+g)*(1-exp(-k*T))/(g*k)};
> k := 0.1e-6;
```

```
> V := 600*sin(3.14159265358979*(1/3));

> g := 9.8;
fsolve(t1,{T=10..120});

                  {T = 106.0398126}


> t2 := {T = (a*V+g)*(1-exp(-a*T))/(g*a)};
> a := 0.005;

> V := 600*sin(3.14159265358979*(1/3));
> g := 9.8;

> fsolve(t2, {T = 10 .. 120});

                  {T = 98.06233448}


> t3 := {T = (b*V+g)*(1-exp(-b*T))/(g*b)};
> b := 0.01;

> V := 600*sin(3.14159265358979*(1/3));
> g := 9.8;

> fsolve(t3, {T = 10 .. 120});

                  {T = 92.10191668}
```

and so on for the various values of k.

PYTHON

```
# Range Calculator for different drag coefficients

# Edit y = 0 range and t increment to adjust accuracy (near the end of the
while loop)



from math import *


# constants

g = 9.8

pi = 3.141592653589793

theta = 60*(pi/180)

v0 = 600

u = v0*cos(theta)

v = v0*sin(theta)


# variables to play with

t = 0

y = 1


# some blank lists to get started

R = []

kl = []


# generate drag coefficients and us n to iterate through them

k = range(1,40,1)

n = range(0,len(k),1)
```

```
for i in n:

   while t <= 130 and y >= 0:

      x = (u/(k[i]*0.001)*(1-exp((-(k[i]*0.001)*t))))

      y = (-g)*(t/(k[i]*0.001))+(((k[i]*0.001)*v+g)/((k[i]*0.001)**2))*(1-exp((-
(k[i]*0.001))*t))

      x = round(x,2)

      y = round(y,2)

      if 0 < y < 25: # if particle is within 25 meters of the ground, save its x
coordinate

         R.append(x)

         kl.append(k[i]*0.001) # also save the k value corresponding to that
saved range

         print(x,y)

      t += 0.1

   x,y,t = 0,0,0  # reset x, y and t to origin


print(R)

print(kl)
```